MODIFIKASI *STEPSIZE* PADA METODE *STEEPEST DESCENT* DALAM PENGOPTIMUMAN FUNGSI: KASUS FUNGSI KUADRATIK DIAGONAL

F. FADHILLAH[1], B. P. SILALAHI[2], M. ILYAS[2]

Abstrak

Metode *steepest descent* adalah salah satu metode untuk menemukan titik optimum suatu fungsi tanpa kendala. Metode ini menggunakan *stepsize* yang diperoleh dari pencarian *exact line*. Metode ini mungkin menuju ke titik optimum dengan lambat. Beberapa penelitian telah dilakukan untuk mengatasi kelemahan ini dengan mengubah *stepsize*. Beberapa *stepsize* baru antara lain dikembangkan oleh Ya-xiang Yuan, Barzilai, dan Borwein. Penelitian ini membandingkan waktu penyelesaian dan banyaknya iterasi untuk ketiga metode disebut di atas dalam menyelesaikan suatu permasalahan pengoptimuman tanpa kendala untuk kasus fungsi kuadratik diagonal. Hasil numerik yang diperoleh menunjukan bahwa metode Ya-xiang Yuan dapat menemukan titik optimum hanya dengan tiga iterasi saja untuk fungsi dengan dua variabel. Selanjutnya metode Ya-xiang Yuan sangat efisien untuk masalah dengan dimensi kecil, sedangkan metode Barzilai-Borwein menunjukan hasil yang lebih baik untuk masalah dengan dimensi besar.

Kata kunci: Modifikasi *stepsize*, pengoptimuman fungsi tanpa kendala, *Steepest descent*

1 PENDAHULUAN

Latar Belakang

Permasalahan mengenai pengoptimuman adalah mencari penyelesaian terbaik dari suatu masalah. Masalah yang ditemui terdiri atas fungsi tujuan dan kendala yang dapat berupa fungsi linear maupun nonlinear.

Terdapat beberapa metode untuk menyelesaikan masalah pengoptimuman dengan kelebihan dan kekurangan yang berbeda untuk masing-masing metode. Salah satu metode yang digunakan dalam masalah pengoptimuman bersifat iteratif, yaitu dimulai dari titik awal x_1 yang sudah ditentukan, kemudian bergerak ke titik x_2 hingga titik x_n, yaitu titik yang mendekati atau sama dengan nilai optimal. Metode-metode tersebut dapat diklasifikasikan ke dalam dua kelompok, yaitu metode dengan menggunakan gradien dan metode tanpa menggunakan gradien.

[1]Mahasiswa Program Sarjana, Departemen Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Jalan Meranti Kampus IPB Dramaga Bogor, 16680.
[2]Departemen Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Jalan Meranti Kampus IPB Dramaga Bogor, 16680.

Untuk metode dengan menggunakan gradien, diperlukan syarat fungsi tujuan terturunkan. Contoh metode dengan menggunakan gradien yaitu metode *steepest descent*, metode Newton, dan metode *conjugate* gradien. Contoh metode tanpa menggunakan gradien yaitu metode Rosenbrock dan metode Nelder-Mead [2]. Masalah yang digunakan pada penelitian ini adalah masalah pengoptimuman nonlinear tanpa kendala, yaitu mencari nilai x yang meminimumkan suatu fungsi $f(x)$ dan metode yang digunakan adalah metode *steepest descent.*

Metode *steepest descent* bergerak dengan langkah-langkah yang saling tegak lurus. Tepatnya, jika $\{x_i\}$ adalah barisan *steepest descent* untuk fungsi $f(x)$, maka untuk setiap bilangan asli $i \geq 0$, vektor yang menghubungkan x_{i-1} dengan x_i tegak lurus dengan vektor yang menghubungkan x_i dengan x_{i+1}.

Perlu diketahui, pencarian dengan arah *steepest descent* untuk menuju ke suatu titik mungkin berjalan dengan lambat [3]. Metode *steepest descent* memerlukan iterasi yang banyak untuk menemukan solusi minimum karena gerak langkahnya yang berliku-liku (*zigzag*). Barzilai dan Borwein [1] berusaha menyempurnakan metode ini dengan mengubah *stepsize*. Metode ini kemudian dikenal dengan metode BB.

Pada penelitian ini akan dibahas tentang modifikasi metode *steepest descent* dengan mengubah *stepsize*. Algoritme yang baru ini akan menempatkan *stepsize* yang baru pada iterasi genap sedangkan pada setiap iterasi ganjil tetap menggunakan *stepsize* pada *steepest descent*. Algoritme ini didasarkan pada artikel yang ditulis oleh Yuan [3]. Setelah itu akan dilakukan simulasi sebagai perbandingan dengan metode *steepest descent* dan metode BB. Pengolahan data dilakukan dengan menggunakan bantuan *software* MATLAB R2010a.

Tujuan Penelitian

Penelitian ini bertujuan untuk:

1. Merekonstruksi penggunaan *stepsize* baru pada metode *steepest descent.*
2. Membandingkan waktu penyelesaian dan banyaknya iterasi yang dilakukan pada metode *steepest descent* yang telah dimodifikasi, Barzilai dan Borwein, dan *steepest descent*.

2 TINJAUAN PUSTAKA

Metode *steepest descent* adalah metode gradien sederhana untuk pengoptimuman tanpa kendala:

$$\min_{x \in \mathbb{R}^n} f(\mathbf{x}),$$

dengan $f(\mathbf{x})$ adalah fungsi kontinu dan terturunkan di $\mathbb{R}^n$, yaitu fungsi $f(\mathbf{x})$ kontinu dan terturunkan. Metode ini memiliki bentuk sebagai berikut:

$$x_{k+1} = x_k + \alpha_k(-g_k),$$

dengan $g_k = g(x_k) = \nabla f(x_k)$ adalah vektor gradien dari $f(\mathbf{x})$ di x_k dan $\alpha_k > 0$ adalah *stepsize.* Arah pencarian dalam metode ini berbanding terbalik dengan arah gradien, yaitu dengan arah menurun tercuram (s*teepest descent*), sehingga metode ini diberi nama *steepest descent*. Arah curam menurun yang diterapkan dalam metode ini sendiri dipercaya merupakan arah terbaik dalam artian metode ini dapat mengurangi fungsi objektif sebanyak mungkin.

Stepsize α_k dapat diperoleh dengan pencarian *exact line*:

$$\alpha_k = \text{argmin}\{f(x_k + a(-g_k))\}.$$

Metode s*teepest descent* selalu konvergen. Secara teori metode ini tidak akan berhenti atau akan terus melakukan iterasi sampai kriteria penghentian terpenuhi. Namun, untuk kasus yang sangat sederhana saat fungsi objektif $f(\mathbf{x})$ merupakan fungsi kuadrat konveks sempurna, yaitu:

$$f(\mathbf{x}) = g^T x + \frac{1}{2} x^T H x,$$

dengan $g \in \mathbb{R}^n, H \in \mathbb{R}^{n \times n}$ simetris dan definit positif. Asumsikan kita menggunakan *stepsize* yang didapat dari pencarian *exact line,* metode ini dapat membutuhkan waktu yang cukup lama untuk memperoleh hasil [3].

3 HASIL DAN PEMBAHASAN

Metode Ya-xiang Yuan

Untuk analisis pada bab ini, diasumsikan bahwa fungsi objektif adalah sebagai berikut:

$$f(x) = g^T x + \tfrac{1}{2} x^T H x,$$

dengan $g \in \mathbb{R}^n$ dan $H \in \mathbb{R}^{n \times n}$ simetris dan definit positif. Pada dasarnya, metode baru ini adalah pengembangan dari metode *steepest descent*. Dapat dilihat pada metode baru ini pencarian *exact line* harus dilakukan pada iterasi terakhir sebelum algoritme berhasil menemukan solusinya. Diasumsikan pula bahwa digunakan pencarian *exact line* pada iterasi pertama supaya kita tidak menghindari keberuntungan apabila ada kasus di mana algoritme dapat menemukan solusi pada iterasi pertama. Oleh karena itu, dibuatlah algoritme sebagai berikut:

$$x_2 = x_1 - \alpha_1^* g_1$$
$$x_3 = x_2 - \alpha_2 g_2$$
$$x_4 = x_3 - \alpha_3^* g_3,$$

di mana α_1^* dan α_3^* didapat dari pencarian *exact line* dan x_4 adalah solusi. Perlu dicari formula untuk α_2 sehingga x_4 akan menjadi nilai minimum dari fungsi objektif. Metode ini disebut metode Ya-xiang Yuan.

Untuk mempermudah analisis, dipelajari kasus di mana g_1 dan g_2 adalah dua buah sumbu. Sesuai dengan pencarian *exact line* pada iterasi pertama, gradien g_1 dan g_2 adalah ortogonal. Oleh karena itu kita dapat menampilkan semua vektor x sebagai kombinasi linear dari g_1 dan g_2. Misalkan diberikan fungsi:

$$f\left(x_2 + t\frac{g_1}{\|g_1\|} + u\frac{g_2}{\|g_2\|}\right) = \begin{pmatrix} 0 \\ \|g_2\| \end{pmatrix}^T \begin{pmatrix} t \\ u \end{pmatrix} + {}^1\!/_2 \begin{pmatrix} t \\ u \end{pmatrix}^T \begin{pmatrix} g_1^T H g_1 / \|g_1\|^2 & g_1^T H g_2 / \|g_1\|\|g_2\| \\ g_1^T H g_2 / \|g_1\|\|g_2\| & g_2^T H g_2 / \|g_2\|^2 \end{pmatrix} \begin{pmatrix} t \\ u \end{pmatrix}.$$

Berdasarkan pencarian *exact line* pada iterasi pertama, diperoleh $\alpha_1^* = \|g_1\|^2 / g_1^T H g_1$ dan $g_1^T H g_2 = -\|g_2\|^2 / \alpha_1^*$ yang diperoleh dari:

$$x_{k+1} = x_k - \alpha_k^* g_k$$

$$f(x_{k+1)} = f(x_k - \alpha_k^* g_k)$$

$$f'(x_{k+1}) = f'(x_k - \alpha_k^* g_k).$$

Karena $\alpha_k^* = \text{argmin}\,\{f(x_k - \alpha_k^* g_k)\}$, diperlukan syarat $f'(x_{k+1}) = 0$ sehingga:

$$f'(x_k - \alpha_k^* g_k) = 0$$

$$-f'(x_k - \alpha_k^* g_k)^{\,T} g_k = 0$$

$$-\big(b + H(x_k - \alpha_k^* g_k)\big)^T (b + Hx) = 0$$

$$-(b + H_x - \alpha_k^* H(g_k)^T (b + Hx) = 0$$

$$-(b + Hx)^T (b + Hx) + \alpha_k^* (b + Hx)^T H (b + Hx) = 0$$

$$-g_k^T g_k + \alpha_k^* g_k^T H g_k = 0$$

$$\alpha_k^* g_k^T H g_k = g_k^T g_k$$

$$\alpha_k^* = \frac{g_k^T g_k}{g_k^T H g_k}$$

$$\alpha_k^* = \frac{\|g_k\|^2}{g_k^T H g_k}.$$

Dengan menggunakan notasi $\alpha_2^* = \|g_2\|^2 / g_2^T H g_2$, didapat bahwa:

$$f(x_2 + t\frac{g_1}{\|g_1\|} + u\frac{g_2}{\|g_2\|}) = \begin{pmatrix} 0 \\ \|g_2\| \end{pmatrix}^T \begin{pmatrix} t \\ u \end{pmatrix} + {}^1\!/_2 \begin{pmatrix} t \\ u \end{pmatrix}^T \begin{pmatrix} {}^1\!/_{a_1^*} & -\|g_2\|/a_1^*\|g_1\| \\ -\|g_2\|/a_1^*\|g_1\| & {}^1\!/_{a_2^*} \end{pmatrix} \begin{pmatrix} t \\ u \end{pmatrix}.$$

Oleh karena itu, dapat diketahui nilai minimum dari fungsi objektifnya adalah:

$$\begin{pmatrix} t^* \\ u^* \end{pmatrix} = -\frac{\|g_1\|\|g_2\|}{\|g_1\|^2/a_2^* - \|g_2\|^2/a_1^*} \begin{pmatrix} \|g_2\| \\ \|g_1\| \end{pmatrix},$$

yang diperoleh dari:

$$\frac{df}{dt} = 0$$

$$\frac{2t}{a_1^*} - \frac{2u\|g_2\|}{a_1^*\|g_1\|} = 0$$

$$2t\|g_1\| - 2u\|g_2\| = 0$$

$$t\|g_1\| - u\|g_2\| = 0 \qquad (1)$$

$$\frac{df}{du} = 0$$

$$\|g_2\| + \frac{u}{a_2^*} - \frac{t\|g_2\|}{a_1^*\|g_1\|} = 0$$

$$-ta_2^*\|g_2\| + ua_1^*\|g_1\| = -a_1^*a_2^*\|g_1\|\|g_2\|. \qquad (2)$$

Kemudian dilakukan eliminasi pada persamaan (1) dan (2)

(1) * $a_2^*\|g_2\|$ $$ta_2^*\|g_1\|\|g_2\| - ua_2^*\|g_2\|^2 = 0 \quad (3)$$

(2) * $\|g_1\|$ $$-ta_2^*\|g_1\|\|g_2\| + ua_1^*\|g_1\|^2 = -a_1^*a_2^*\|g_1\|^2\|g_2\| \quad (4)$$

(3) + (4) $$ua_1^*\|g_1\|^2 - ua_2^*\|g_2\|^2 = -a_1^*a_2^*\|g_1\|^2\|g_2\|$$

$$u = -\frac{a_1^*a_2^*\|g_1\|^2\|g_2\|}{a_1^*\|g_1\|^2 - a_2^*\|g_2\|^2}$$

$$u = -\frac{\|g_1\|^2\|g_2\|}{\|g_1\|^2/a_2^* - \|g_2\|^2/a_1^*}.$$

Subtitusikan u ke persamaan (1) diperoleh:

$$t = -\frac{\|g_1\|\|g_2\|^2}{\|g_1\|^2/a_2^* - \|g_2\|^2/a_1^*}.$$

Untuk mendapatkan $x_4 = x_2 + t^*\frac{g_1}{\|g_1\|} + u^*\frac{g_2}{\|g_2\|}$, perlu diketahui bahwa arah gradien g_3 paralel terhadap vektor residual $x_4 - x_3$. Untuk itu, diperlukan dua arah:

$$\begin{pmatrix} t^* \\ u^* \end{pmatrix} - \begin{pmatrix} 0 \\ -\alpha_2\|g_2\| \end{pmatrix}$$

dan

$$\begin{pmatrix} 0 \\ \|g_2\| \end{pmatrix} + \begin{pmatrix} 1/a_1^* & -\|g_2\|/a_1^*\|g_1\| \\ -\|g_2\|/a_1^*\|g_1\| & 1/a_2^* \end{pmatrix}\begin{pmatrix} 0 \\ -\alpha_2\|g_2\| \end{pmatrix}$$

adalah dua buah arah yang paralel. Dua arah tersebut paralel terhadap masing-masing:

$$\begin{pmatrix} \|g_2\| \\ \|g_1\| - \alpha_2(\frac{\|g_1\|^2}{\alpha_2^*} - \frac{\|g_2\|^2}{\alpha_1^*})/\|g_1\| \end{pmatrix}$$

dan

$$\begin{pmatrix} \alpha_2\|g_2\|/\alpha_1^*\|g_1\| \\ 1 - \alpha_2/\alpha_2^* \end{pmatrix}.$$

Dapat diasumsikan:

$$\begin{pmatrix} \|g_2\| \\ \|g_1\| - \alpha_2(\frac{\|g_1\|^2}{\alpha_2^*} - \frac{\|g_2\|^2}{\alpha_1^*})/\|g_1\| \end{pmatrix} = \lambda \begin{pmatrix} \alpha_2\|g_2\|/\alpha_1^*\|g_1\| \\ 1 - \alpha_2/\alpha_2^* \end{pmatrix}$$

untuk $\lambda \in \Re$. Berdasarkan baris pertama didapatkan $\lambda = \alpha_1^*\|g_1\|/\alpha_2$. Kemudian nilai λ tersebut disubtitusikan ke baris kedua pada persamaan di atas sehingga diperoleh:

$$1 - \alpha_2 \left(\frac{1}{\alpha_2^*} - \frac{\|g_2\|^2}{\alpha_1^*\|g_1\|^2}\right) = \frac{\alpha_1^*}{\alpha_2^*} - \frac{\alpha_1^*}{\alpha_2^*}.$$

Persamaan di atas ekuivalen dengan:

$$\left(\frac{1}{\alpha_1^*\alpha_2^*} - \frac{\|g_2\|^2}{(\alpha_1^*\|g_1\|)^2}\right)\alpha_2^2 - \left(\frac{1}{\alpha_1^*} - \frac{1}{\alpha_2^*}\right)\alpha_2 + 1 = 0. \qquad (3)$$

Karena H definit positif, diketahui bahwa:

$$\Gamma = \frac{1}{\alpha_1^*\alpha_2^*} - \frac{\|g_2\|^2}{(\alpha_1^*\|g_1\|)^2} > 0.$$

Dari persamaan (3) diperoleh dua solusi positif untuk α_2 yaitu:

$$\frac{(1/\alpha_1^* + 1/\alpha_2^*) \pm \sqrt{(1/\alpha_1^* + 1/\alpha_2^*)^2 - 4\Gamma}}{2\Gamma}.$$

Dari dua solusi tersebut dipilih nilai yang lebih kecil dan dapat dituliskan sebagai berikut:

$$\alpha_2 = \frac{2}{\sqrt{(1/\alpha_1^* - 1/\alpha_2^*)^2 + 4\|g_2\|^2/\|s_1\|^2 + 1/\alpha_1^* + 1/\alpha_2^*}},$$

dengan $s_1 = x_2 - x_1 = -\alpha_1^* g_1$. α_2 inilah yang disebut *stepsize* baru dan kemudian akan diaplikasikan ke dalam metode hasil modifikasi *steepest descent.*

Metode Barzilai Borwein

Gagasan utama dari metode Barzilai dan Borwein ini adalah dengan menggunakan hasil pada iterasi sebelumnya untuk menentukan *stepsize.* Iterasi yang digunakan adalah sebagai berikut:

$$x_{k+1} = x_k - D_k g_k \ ,$$

di mana $D_k = \lambda_k I$. Metode ini memiliki dua buah *stepsize*:

$$\lambda_k = \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T y_{k-1}}$$

dan

$$\lambda_k = \frac{s_{k-1}^T y_{k-1}}{y_{k-1}^T y_{k-1}},$$

dengan $s_{k-1} = x_k - x_{k-1}$ dan $y_{k-1} = g_k - g_{k-1}$. Pada penelitian ini hanya digunakan satu buah *stepsize* yaitu:

$$\lambda_k = \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T y_{k-1}}.$$

Hasil Numerik

Algoritme yang digunakan pada penelitian ini adalah sebagai berikut:

Algoritme Ya-xiang Yuan

Step 1 Masukan titik awal x_1. $0 < \varepsilon \ll 1$. Hitung g_1, Tetapkan k=1.
Step 2 Hitung stepsize dengan pencarian exact line α_{2k-1}^*; Tetapkan

$$x_{2k} = x_{2k-1} - \alpha_{2k-1}^* g_{2k-1}$$

Step 3 Jika $\|g(x_{2k})\| \leq \varepsilon$ maka berhenti;
Step 4 Hitung stepsize dengan pencarian exact line α_{2k}^* , Tetapkan

$$\alpha_{2k} = \frac{2}{\sqrt{(1/\alpha_{2k-1}^* - 1/\alpha_{2k}^*)^2 + 4\|g_{2k}\|^2/\|s_{2k-1}\|^2} + 1/\alpha_{2k-1}^* + 1/\alpha_{2k}^*}$$

dan

$$x_{2k+1} = x_{2k} - \alpha_{2k} g_{2k}$$

Jika $\|g_{2k+1}\| \leq \varepsilon$ maka berhenti;
Step 5 k=k+1, kembali ke Step 2.

Algoritme *Steepest Descent*

Step 1 Masukan titik awal x_0. $0< \varepsilon \ll 1$. Hitung g_1, Tetapkan k=0.
Step 2 Hitung stepsize dengan pencarian exact line α_k^*; Tetapkan

$$x_{k+1} = x_k - \alpha_k^* g_k$$

Step 3 Jika $\|g(x_k)\| \leq \varepsilon$ maka berhenti;
Step 4 k=k+1, kembali ke Step 2.

Algoritme Barzilai-Borwein

Step 1. Diberikan $x_0 \in \mathbb{R}^n$, $0< \varepsilon \ll 1$. Tetapkan k=0.
Step 2. Jika $\| g_k \| \leq \varepsilon$, stop; lainnya $d_k = -g_k$.
Step 3. Jika k=0, menentukan λ_0 dengan pencarian exact line; selainnya dengan

menghitung λ_k dengan $\lambda_k = \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T y_{k-1}}$.

Step 4. Tentukan $x_{k+1} = x_k + \lambda_k d_k$.
Step 5. $k = k + 1$, kembali ke Step 2.

Fungsi yang digunakan adalah fungsi kuadratik diagonal, yaitu fungsi yang dibangkitkan secara acak dengan ketentuan sebagai berikut:

$$f(x) = (x - x^*)^T Diag(\sigma_1, \dots, \sigma_n)(x - x^*). \quad x \in \Re^n.$$

Jumlah variabel yang digunakan dilambangkan dengan n di mana nilai n =2,3,10,25. Vektor $x_i^*(i = 1, \dots, n) \in (-5,5)$ yang dipilih secara acak. Diberikan $\sigma_n = Cond(10,100)$ dan $\sigma_i(i = 1, \dots, n)$ di mana nilainya diperoleh secara acak dengan interval $(1, \sigma_n)$. Untuk semua kasus diberikan titik awal adalah vektor nol $(0, \dots, 0)^T$ dan kriteria penghentian adalah $\|g_k\| \leq 10^{-6}$. Untuk setiap kasus, dilakukan lima kali pengulangan.

Tabel 1
Hasil untuk fungsi dua variabel

		Ya-xiang Yuan		*Steepest Descent*		BB	
n	σ_n	Iterasi	Waktu (s)	Iterasi	Waktu (s)	Iterasi	Waktu (s)
2	10	3	0,166250	14	0,952635	5	0,265793
		3	0,164151	6	0,474521	5	0,272782
		3	0,167652	6	0,441272	5	0,289684
		3	0,163253	15	0,993925	9	0,406331
		3	0,155448	16	1,070552	7	0,345200
	100	3	0,191924	5	0,391837	5	0,261558
		3	0,161772	25	1,644183	13	0,535388
		3	0,157719	8	0,592578	7	0,342918
		3	0,159453	5	0,392321	5	0,268820
		3	0,156356	8	0,569770	5	0,269873
Rata-rata		3	0,164398	10,8	0,752359	6,6	0,325835

Untuk fungsi dengan dua variabel, metode Ya-xiang Yuan hanya membutuhkan maksimal tiga iterasi untuk menemukan solusi minimumnya (Tabel 1). Metode ini memiliki waktu yang paling cepat dan banyaknya iterasi paling sedikit dibandingkan metode BB dan *steepest descent.*

Tabel 2
Hasil untuk fungsi tiga variabel

		Ya-xiang Yuan		*Steepest Descent*		BB	
n	σ_n	Iterasi	Waktu (s)	Iterasi	Waktu (s)	Iterasi	Waktu (s)
3	10	8	0,352896	12	0,889177	9	0,487921
		6	0,288516	55	3,814177	13	0,631213
		7	0,320067	54	3,790647	17	0,788778
		3	0,170093	9	0,693719	5	0,320157
		12	0,523204	22	1,599422	14	0,681257
	100	8	0,399514	18	1,351256	9	0,483705
		11	0,470182	102	6,948871	15	0,717084
		11	0,493538	36	2,514216	13	0,651547
		8	0,353545	30	2,144351	15	0,714584
		9	0,407075	26	1,884509	9	0,484089
Rata-rata		8,3	0,371715	36,4	2,426633	11,9	0,596034

Untuk fungsi dengan tiga variabel, metode Ya-xiang Yuan memiliki waktu yang paling cepat dan banyaknya iterasi paling sedikit dibandingkan metode BB dan *steepest* descent. Pada salah satu percobaan, metode Ya-xiang Yuan dan metode BB menghasilkan banyak iterasi yang sama yaitu sebesar sembilan iterasi, namun waktu yang dibutuhkan metode Ya-xiang Yuan lebih cepat daripada metode BB (Tabel 2). Berdasarkan nilai rata-rata, metode Ya-xiang Yuan dan BB hanya

membutuhkan waktu kurang dari satu sekon untuk menemukan solusi, sedangkan metode *steepest descent* membutuhkan waktu lebih dari dua sekon.

Tabel 3
Hasil untuk fungsi sepuluh variabel

		Ya-xiang Yuan		*Steepest Descent*		BB	
n	σ_n	Iterasi	Waktu (s)	Iterasi	Waktu (s)	Iterasi	Waktu (s)
10	10	23	2,349185	34	5,079649	23	2,055472
		19	2,016547	33	3,560591	17	1,670175
		17	1,745476	26	2,741951	17	1,638760
		15	1,614203	21	2,308352	13	1,369932
		20	2,062425	36	3,897925	18	1,682789
	100	27	2,821865	63	6,637803	25	2,430587
		36	3,617917	75	7,750232	26	2,311006
		65	6,452713	311	31,104918	45	3,758671
		29	3,001316	49	5,749044	27	2,413808
		35	3,413009	65	7,332228	31	2,699278
Rata-rata		28,6	2,909466	71,3	7,898116	24,2	2,265746

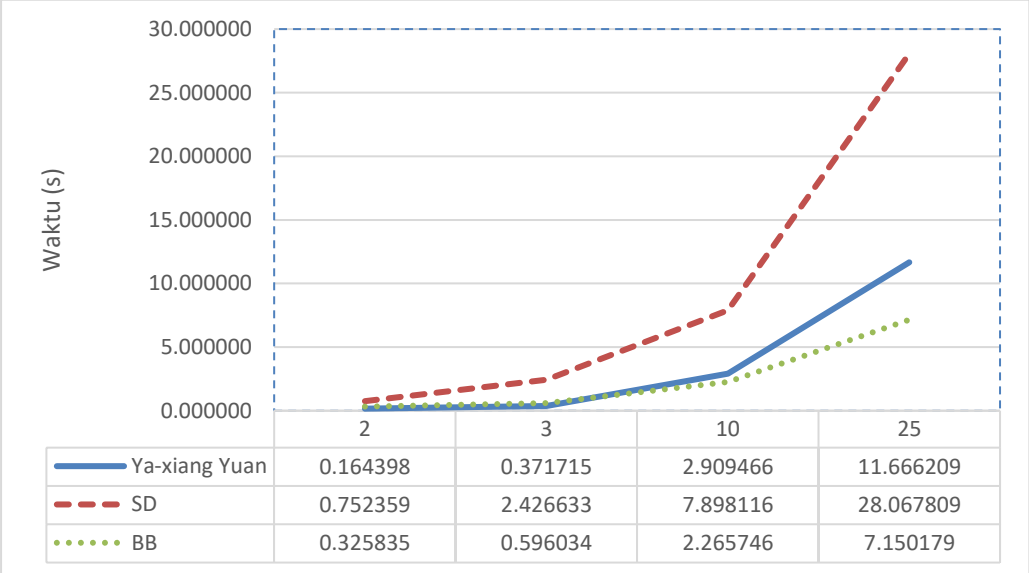
Untuk fungsi dengan sepuluh variabel, metode BB memiliki waktu yang paling cepat dan banyaknya iterasi paling sedikit dibandingkan metode Ya-xiang Yuan dan *steepest descent*. Pada salah satu percobaan, metode Ya-xiang Yuan dan metode BB menghasilkan banyaknya iterasi yang sama yaitu sebesar 17 iterasi, namun waktu yang dibutuhkan metode BB lebih cepat daripada metode Ya-xiang Yuan (Tabel 3). Berdasarkan nilai rata-rata, metode BB membutuhkan waktu sekitar dua sekon untuk menemukan solusi minimumnya, metode Ya-xiang Yuan membutuhkan waktu hampir tiga sekon, dan metode *steepest descent* membutuhkan waktu sekitar tujuh sekon.

Tabel 4
Hasil untuk fungsi 25 variabel

		Ya-xiang Yuan		*Steepest Descent*		BB	
n	σ_n	Iterasi	Waktu (s)	Iterasi	Waktu (s)	Iterasi	Waktu (s)
25	10	33	8,323247	70	12,596562	33	5,938817
		31	7,952204	64	11,993606	27	5,167664
		31	8,029839	68	12,730917	27	5,266595
		33	8,415656	72	13,408431	29	5,481138
		33	9,730223	74	13,550629	29	5,522943
	100	62	15,998245	216	40,280921	53	9,297413
		45	11,524196	132	24,407456	35	6,487686
		39	10,020371	170	31,759268	37	6,798567
		81	20,716241	304	61,059319	69	11,826102
		79	20,283904	314	58,890976	55	9,714863
Rata-rata		46,7	11,666209	148,4	28,067809	39,4	7,150179

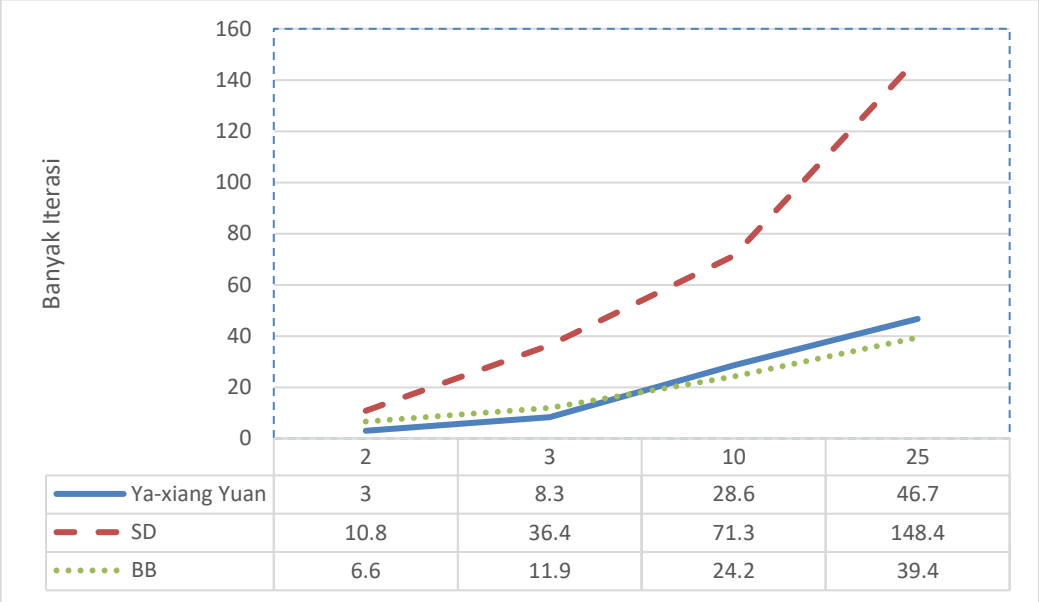
Untuk fungsi dengan 25 variabel, metode BB memiliki waktu yang paling cepat dan banyaknya iterasi paling sedikit dibandingkan metode Ya-xiang Yuan dan *steepest descent*. Pada salah satu percobaan, metode Ya-xiang Yuan dan metode BB menghasilkan banyaknya iterasi yang sama yaitu sebesar 33 iterasi, namun waktu yang dibutuhkan metode BB lebih cepat daripada metode Ya-xiang Yuan (Tabel 4). Berdasarkan nilai rata-rata, metode BB membutuhkan waktu sekitar tujuh sekon untuk menemukan solusi minimumnya, metode Ya-xiang Yuan membutuhkan waktu sekitar 11 sekon, dan metode *steepest descent* membutuhkan waktu sekitar 28 sekon

Perbandingan waktu eksekusi dan banyaknya iterasi untuk metode Ya-xiang Yuan, metode BB, dan metode *steepest descent* dalam bentuk grafik, ditunjukkan pada Gambar 1 dan Gambar 2 sebagai berikut:



	2	3	10	25
Ya-xiang Yuan	0.164398	0.371715	2.909466	11.666209
SD	0.752359	2.426633	7.898116	28.067809
BB	0.325835	0.596034	2.265746	7.150179

Gambar 1 Perbandingan waktu metode Ya-xiang Yuan, metode BB, dan metode *steepest descent*



Gambar 2 Perbandingan banyak iterasi metode Ya-xiang Yuan, metode BB, dan metode *steepest descent*

Untuk fungsi dengan dua dan tiga variabel, metode Ya-xiang Yuan memiliki waktu yang paling cepat untuk menemukan solusi dibandingkan metode BB dan *steepest descent*, sedangkan untuk fungsi dengan sepuluh dan 25 variabel, metode BB memiliki waktu yang paling cepat untuk menemukan solusi dibandingkan metode Ya-xiang Yuan dan *steepest descent*. Metode *steepest descent* memiliki waktu yang paling lambat dibandingkan metode BB dan Ya-xiang Yuan baik untuk fungsi dengan dua, tiga, sepuluh, maupun 25 variabel (Gambar 1).

Untuk fungsi dengan dua dan tiga variabel, metode Ya-xiang Yuan memiliki iterasi paling sedikit dibandingkan metode BB dan *steepest descent*, sedangkan untuk fungsi dengan sepuluh dan 25 variabel, metode BB memiliki iterasi paling sedikit dibandingkan metode Ya-xiang Yuan dan *steepest descent*. Metode *steepest descent* memiliki banyaknya iterasi yang paling besar dibandingkan metode BB dan Ya-xiang Yuan baik untuk fungsi dengan dua, tiga, sepuluh, maupun 25 variabel (Gambar 2).

4 SIMPULAN

Modifikasi dengan memberikan *stepsize* baru yang dilakukan pada metode Ya-xiang Yuan dapat menemukan solusi suatu masalah nonlinear tanpa kendala dengan waktu yang lebih cepat dan jumlah iterasi yang lebih sedikit dibandingkan metode BB dan metode *steepest descent* untuk masalah dimensi kecil. Hasil

percobaan bahkan menunjukan bahwa metode Ya-xiang Yuan dapat menemukan nilai minimum dengan tiga iterasi saja untuk fungsi dua variabel. Untuk masalah dengan dimensi yang besar, metode BB memberikan kinerja yang lebih baik dibandingkan metode Ya-xiang Yuan maupun metode *steepest descent.*

DAFTAR PUSTAKA

[1] Barzilai J, Borwein JM. 1988. Two point step size gradient methods. *IMA J Numer Anal* 8(1): 141-148. doi:10.1093/imanum/8.1.141.
[2] Klerk E, Roos C, Terlaky T. 2005. *Optimization*. Delft(ND): Delft University of Technology.
[3] Yuan Y. 2006. A new stepsize for the steepest descent method. *Journal of Computational Mathematics* 24(2): 149-156.